### Twig

Objetivo del proyecto

* Entender que es un motor de templates
* Entender la sintaxis de **TWIG** y ponerla en práctica
* Desarrollar un proyecto que ponga en práctica los PROS de usar sistema de plantillas.
* Mejorar tus habilidades en arquitectura, separación de responsabilidades, y buenas prácticas.

### Analiza la interfaz

El primer punto de este proyecto es crear un script escrito en **PHP** para incorporar posteriormente **TWIG.**Crea un repositorio con el siguiente código:

* Fichero index.php como punto de entrada a tu aplicación web
* Crea un directorio llamado **templates** para poder almacenar tus vistas que posteriormente **TWIG** renderizará.
* Define un array multidimensional para almacenar un listado de noticias básico:
  + Título de la noticia
  + Fecha de la noticia en formato DateTime
  + Breve descripción de la noticia

### Instalamos Twip mediante composer

Requisitos del proyecto

Instalamos Twig como requisitos mediante composer en el fichero PHP

* FilesystemLoader
* Environment

El trabajo con las vistas

Crearemos una vista y templates necesario para trabajar en el proyecto

Las siguientes necesidades

* Template o vista primaria llamada: base.html.twig
* Debe contener 4 bloques. Entender para que sirve estos bloques en TWIG. Será mas fácil definirlos
* Template home, será la renderización del script. Esta vista será heredada de base.html.twig (es como herencia entre vista)
* Se Creara un formulario de contacto que hago uso de una macro previamente definida que se encargara de renderizar los campos de texto input. Será parametrizable para indicar el tipo input ( text, number, email, password, etc ).
* Se debe iterar las noticias para mostrar el listado de la misma.
* Es importante que hagas una sentencia de repetición responsable de mostrar una única noticia.
* Comprobar si existen nioticias tanto como variable o como numero.
* La vista debe tener otro comportamiento en caso que no existan noticias

Ejecutador el método Render

Este método se encargar de obtener el resultado de la vista HTML

La vista es capaz de renderizar las noticias es necesario que estas sean un parámetro del método render.

Implementación del proyecto



|  |  |  |  |
| --- | --- | --- | --- |
| **Tarea** | **Prioridad** | **Horas** | **Dificultad** |
| Documentación | Normal | 1,00 | Alta |
| Organización | Alta | 1,00 | Alta |
| Búsqueda Previa de información | Normal | 1,00 | Normal |
| Creación de repositorio | Baja | 0,15 | Baja |
| Documentando Proyecto | Normal | 1,00 | Normal |
| Previa investigación  Composer | Normal | 1,00 | normal |
| Installar Composer | Alta | 1,00 | Alta |
| Estructura Index.php principal | Normal | 1,00 | Normal |
| Importacion Librería necesarias | Normal | 1:00 | Normal |
| Vistas primarias  base.html.twig | Normal | 1,00 | Normal |
| Bloques Template home | Alta | 2.00 | Alta |
| Bloques formulario de contacto | Alta | 2.00 | Normal |
| Bloques iterar las noticias | Alta | 2.00 | Alta |
| Comprobar que existen noticias | Alta | 2.00 | Normal |
| Creación README | Baja | 2,00 | Baja |
| Testing / Corrección Errores | Alta | 2,00 | Alta |

Tenemos previsto terminar el proyecto en 21 Horas

En las cual damos 8 horas más para prever incidencias en el proyecto

## Especificaciones del proyecto

1. Debes de usar GIT desde el inicio del proyecto
2. Debes desarrollar el proyecto con **PHP**y **WampServer**
3. **Utilizamos Composer para instalar Twig**
4. Deberás usar en nuestro Script **Twig**para importar los componentes necesario
5. Componentes
6. Todo el código debe de estar debidamente documentado
7. Todos los comentarios incluidos en el código tienen que estar escritos en inglés
8. Usa el estilo de código camelCase
9. Es recomendado dividir las tareas en varias subtareas para que de esta forma puedas asociar cada paso en particular de la construcción con un commit específico
10. Para la documentación del proyecto es requerida una versión en PDF dentro del repositorio
11. Debes intentar en la medida de lo posible que los commits y las tareas planificadas sean lo mismo
12. Borra los ficheros que no se use

Registro De Incidencia que se ha detectado durante el proyecto

* Ninguno

Seguimiento Calendario del proyecto

Métricas de Calidad

A pesar de que el proyecto debe adherirse a todos los [*requisitos*](https://docs.google.com/document/d/1dtnHybW7qhBAuuuNEZi1Pb0neJEqbME-hXvGSw2XHUI/edit#heading=h.rf0k2roeckt1) y [*especificaciones*](https://docs.google.com/document/d/1dtnHybW7qhBAuuuNEZi1Pb0neJEqbME-hXvGSw2XHUI/edit#heading=h.xr3uz2kyop9d)del proyecto, hay algunas condiciones que, si se cumplen adecuadamente, añaden una sensación de calidad y robustez al propio proyecto. Estas condiciones son:

1. Levantar el npm server conectándose a mysql y localhost.
2. El código NodeJS utilizamos Visual Studio Code
3. El código NodeJS debe estar libre de pelusas.
4. La aplicación web debe responder.
5. La aplicación web debe ser compatible con los principales navegadores del mercado:
   * Internet Explorer 11 o superior.
   * Safari en una de sus últimas versiones.
   * Firefox en una de sus últimas versiones.
   * Chrome en una de sus últimas versiones.

Documentación de Riesgos

* Retrasos de proyecto.
* Pérdida o daños de material de trabajo.

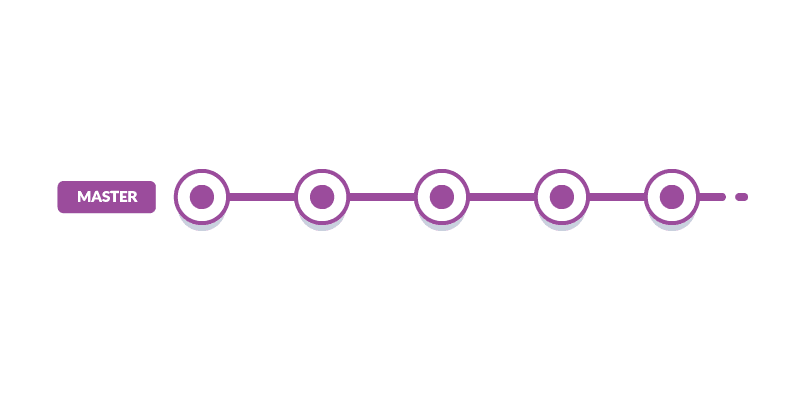
Documentación WORKFLOW DE GIT

* Creación Git Hub https://github.com/robertfox11/NodeJS\_PortFolio.git
* Hacemos commits de la estructura de la página principal.
* Probabilidad de que ocurra 80%
* Impacto en el proyecto 60%
* Posible alternativa (mitigación) Pedir ayuda a compañeros
* Probabilidad de que ocurra 30%
* Impacto en el proyecto 60%
* Posible alternativa (mitigación) Pedir ayuda a compañeros
* No encontrar con facilidad la información relacionada con el proyecto
* Probabilidad de que ocurra 30%
* Impacto en el proyecto 60%
* Posible alternativa (mitigación)
* Pedir ayuda a compañeros

A partir de la realización de la estructura se continuó trabajando solamente en la

rama “master”, a través del Workflow “Gitflow”.

Mas información --> <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow>



Herramiento del proyecto

Se utilizaron diferentes herramientas en el desarrollo del proyecto. Son los siguientes:

1. ***git: un potente sistema de control de*** versiones que ayuda a realizar un seguimiento de los cambios en el árbol de trabajo.
2. ***Visual Studio Code: un editor de*** código optimizado para crear y depurar aplicaciones web modernas.
3. ***Wamps Server, para realizar la el proyecto***
4. ***Composer para hacer las peticiones de las librería***
5. ***Instalacion de TWIG y dependencias necesaria***
6. ***Herramientas para desarrolladores de Google Chrome:*** se utiliza para depurar el código JavaScript y para probar los ajustes de diseño.
7. ***Documentos de Google:*** se utiliza para escribir la documentación del proyecto.
8. [***Validador W3C***](https://validator.w3.org/)***:*** utilizado para validar el código HTML y CSS.

## Flujo de trabajo de Git

Todas las confirmaciones se van a insertar en la rama ***maestra,*** siguiendo un criterio personal de cargar solo instantáneas que son funcionales y que funcionan correctamente, sin contar errores menores. No hay otras ramas, ya que ralentizaría el proceso de desarrollo.

En otro lado, los mensajes de confirmación finalizan con su objetivo principal indicado entre corchetes: por ejemplo.

* Utilizaremos un template index.pug que las instalaremos con npm
* Las relaciones principal para la logica funcional es [class]
* las confirmaciones relacionadas principalmente con los cambios CSS comienzan con ***[styling]***;
* los relacionados principalmente con el diseño de la página, ***[layout]***; los
* Los relacionado con el proyecto añadiendo carpetas o documentacion, plugins, library ***[us]****;*
* relacionados principalmente con la documentación, ***[documentation]***.
* Para NodeJS vista la relacionamos con [Proyecto]
* Esctructura mvc

## Estructura del archivo

Los archivos del proyecto se organizarán de esta manera:

ProyectoNodeJS/

public/

img/ Carpeta que contiene todas las imágenes utilizadas en la interfaz.

js/ Carpeta que contiene todos los scripts utilizados en la interfaz de usuario.

css/ Carpeta que contiene todos los estilos utilizados en la interfaz de usuario.

Config/config.php

Configuración de en rutamiento principal

DataBase.php

Petición de conexión a la base de datos MySql

Controller/

Controladores al modelo y al views

Documentation/

Documentation del proyecto

Helpers/ Una funcionalida de vardump para ver la peticiones

Views/

Vistas de rutas para la acción de los métodos que viene de controllers

/layout

Vistas principales de header footer y aside de pug

.gitignore Carpeta utilizada por *git* para contener información sobre el repositorio.

portafoliobd Usaremos para realizar las peticiones

index.js Página principal de la aplicación web.

README.mdArchivo que contiene instrucciones sobre cómo ejecutar el proyecto.

Registro de lecciones aprendidas.

* Instalaccion de Composer